Unsupervised Batch Normalization

Unsupervised Batch Normalization

Mustafa Taha Koçyiğit, Laura Sevilla-Lara, Timothy M. Hospedales, Hakan Bilen
University of Edinburgh, UK
{taha.kocyigit,l.sevilla,t.hospedales,h.bilen}@ed.ac.uk

Abstract

Batch Normalization is a widely used tool in neural networks to improve the generalization and convergence of training. However, on small datasets due to the difficulty of obtaining unbiased batch statistics it cannot be applied effectively. In some cases, even if there is only a small labeled dataset available, there are larger unlabeled datasets from the same distribution. We propose using such unlabeled examples to calculate batch normalization statistics, which we call Unsupervised Batch Normalization (UBN). We show that using unlabeled examples for batch statistic calculations results in a reduction of the bias of the statistics, as well as regularization leveraging the data manifold. UBN is easy to implement, computationally inexpensive and can be applied to a variety problems. We report results on monocular depth estimation, where obtaining dense labeled examples is difficult and expensive. Using unlabeled samples, and UBN, we obtain an increase in accuracy of more than 6% on the KITTI dataset, compared to using traditional batch normalization only on the labeled samples.

1. Introduction

Large-scale labeled data and modern learnable representations have driven progress on many computer vision problems in the last decade. The dependency on labeled data is a fundamental bottleneck for many vision problems, as obtaining ground truth annotations can either be extremely expensive (*e.g*. pixel-wise annotations for semantic segmentation) or require specialized equipment and controlled environments (*e.g*. depth estimation, 3D pose estimation, optical flow).

In these cases, research often resorts to different strategies. A common approach is to pre-train from scratch a deep network on "realistic" synthetic data and then fine-tune on the available labeled data. This strategy is commonly used for example by state-of-the-art optical flow methods [24]. While this method alleviates overfitting, it requires creating such realistic data, and does not provide the opportunity to leverage the availability of unlabeled data. While data may be difficult and expensive to label, sometimes additional unlabeled data samples (*e.g*. images from the same distribution) can be easily available for free. Another popular technique is transfer learning, where large models are pre-trained on self-supervised tasks (*e.g*. [32]) and then fine-tuned on the small amount of labeled data available. Although the pre-training stage helps the optimization starting point, the fine-tuning stage still risks overfitting to the small amount of labeled data. Another effective strategy is semi-supervised learning [34], which aims at learning both from labeled and unlabeled data at the same time. This approach is successful but is usually not general, and requires specific adaptation to each vision problem.

In this paper we propose a method that is general, does not require adaptation to any new vision problem, leverages additional unlabeled data and does not require fine-tuning. In particular, we adapt the widely used batch normalization (BN) technique to use unlabeled data.

BN is widely used on many architectures to normalize the statistics of the features, and therefore stabilize the optimization process. However, these statistics can also overfit to the labeled dataset and not reflect the statistics that would normalize the actual dataset. This would cause new images to appear as out of distribution examples for the model.

Instead, we use unlabeled samples, which typically are a much larger number than the labeled ones, to compute the normalization statistics, in a process we call *Unsupervised Batch Normalization* (UBN). We test our method in the problem of monocular depth estimation, and obtain more than 6% improvement over the baseline of using vanilla batch normalization. We hope that the simplicity of our method will make it a useful component of future learning systems.

2. Related Work

Normalization methods. Batch normalization [13] is the most standard normalization method in the state-of-the-art classification architectures on the ImageNet LSVRC [25, 30]. Recently, new alternatives have been developed to extend its scope in different ways and making BN more general. Batch Renormalization [12] enables training with

smaller batch sizes and correlated batches by calculating moving averages of batch statistics during training. Mode Norm [6] enables the use of BN for multi-modal data. This is done by calculating multiple batch statistics and adaptively normalizing examples through a gating function and calculating weighted normalization statistic for each example. Instance Norm [28] has been used for style transfer and normalizes the style of the content image [11] by calculating statistics over each channel and sample in the dataset. Adaptive Instance Norm [11] transfers style between two images by calculating the batch statistics form one example and applying them to another image. Weight Norm [22] conditions the optimization problem by reparameterizing the weight values into magnitude and direction while Layer Normalization [2] calculates normalization statistics per example for feature maps in each layer to enable normalization in recurrent models.

While all these variants make BN more usable and stable, to our knowledge, our is the first to extend BN to leverage unlabeled data.

Understanding Batch Normalization. While BN has been widely and successfully used, there remains some questions about why and when it works. The initial explanation of Ioffe and Szegedy [13] that BN reduces internal co-variate shift has been questioned and explored in recent work.

For example, Bjorck et. al. [4] argues that BN enables training with larger learning rates by preventing exploding and vanishing gradients. They also show that unnormalized networks have ill-conditioned activations due to random initialization which results in large singular values of the activation matrices and predictions which are independent from the inputs. Similarly Santurkar et. al. [23] argue that the loss landscape of the normalized networks are smoother. Luo et. al. [18] showed that batch normalization applies population normalization with data dependant gamma decay that promotes uncorrelated features.

Monte Carlo Batch Normalization [27] shows that uncertainty estimates can be obtained from networks with BN layers by sampling batch statistics.

Training on small datasets. Training the classification networks on ImageNet requires millions of balanced labeled images which many other vision problems lack. Many semi-supervised methods that are designed for classification problems [3, 29, 17, 26] are not readily applicable to regression due to the difficulty in generating pseudo-labels and augmentation strategies. Training unsupervised methods like geometric consistency based unsupervised training[16, 9, 1] with supervised training in order to mitigate labeled data requirements has also results in problem specific solutions. While pre-training is a widely used technique in segmentation and detection [5] when the target tasks is not similar to the source tasks the transfer learning has limited success [31]. Another approach is to use synthetic data to pre-train the networks and fine-tune on real data. For tasks like optical flow [24] and depth estimation [15, 19, 10] synthetic data is much more accessible and does not contain the various artifacts and sampling noises that real data has. However, this approach suffers from the domain shift problem where the data distribution of synthetic and real data can differ.

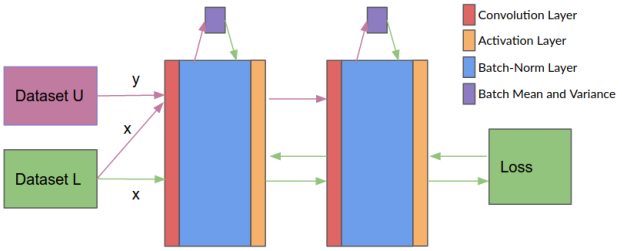

Figure 1. Unsupervised Batch Norm disentangles the normalization from the weight updates by separating them into 2 steps. In the first step a batch is sampled from Dataset U and Dataset L and the normalization statistics are updated. In the second step the sampled batch from Dataset L is used to updated the weights of the network.

3. Method

Since our method is a simple modification of the standard BN method, we start recalling BN and then describe our UBN method.

3.1. Batch Normalization

Given an input batch of height H and width W with N samples and C channels $x \in \mathbb{R}^{N \times C \times H \times W}$, BN normalizes the mean and standard deviation for each individual feature channel c during training:

$$\mathrm{BN}(x)_c = \gamma_c \left(\frac{x_c - \mu(x_c)}{\sigma(x_c)} \right) + \beta_c \tag{1}$$

where $\gamma, \beta \in \mathbb{R}^C$ are affine parameters learned from data; μ, σ are the mean and standard deviation, computed across batch size and spatial dimensions independently for each feature channel.

The moving average of the mean $\hat{\mu}$ and standard deviation $\hat{\sigma}$ are updated using a momentum rate λ during training and used to normalize feature maps during testing.

$$\hat{\mu_c} = \lambda \hat{\mu_c} + (1 - \lambda)\mu(x_c) \tag{2}$$

$$\hat{\sigma_c} = \lambda \hat{\sigma_c} + (1 - \lambda)\sigma(x_c) \tag{3}$$

3.2. Unsupervised Batch Normalization

UBN is based on updating first the batch statistics than the weights. Given an batch $x \in \mathbb{L}$ from labeled dataset $\mathbb{L}$ and a batch $y \in \mathbb{U}$ from unlabeled dataset $\mathbb{U}$ the first step

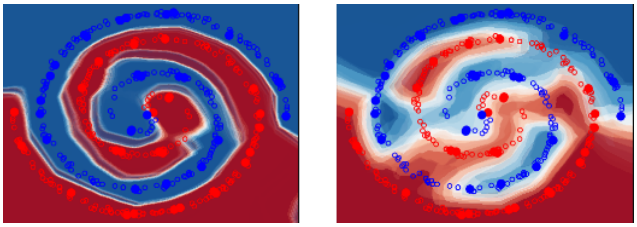

Figure 2. Results of classifying with BN and UBN using the Spiral dataset, with 25 labeled points of each class (filled) and 200 unlabeled points per class which (hollow). Using UBN (left) the model can interpolate between these points while BN (right) learns islands around labeled examples.

builds a combined batch $n = \{x, y\}$ and makes a forward pass to update the normalization statistics. The second step makes forward-backward pass while using x and the updated normalization statistics calculated at the earlier step. 1

UBN normalizes the mean and standard deviation for each individual feature channel with:

$$\text{UBN}(x_c) = \gamma \left(\frac{x_c - \mu(n_c)}{\sigma(n_c)} \right) + \beta \tag{4}$$

where $\mu(n_c), \sigma(n_c)$ are the mean and standard deviation, computed across batch size and spatial dimensions independently for each feature channel from the combined batch n. The $\hat{\mu_c}, \hat{\sigma_c}$ values are calculated similarly to BN using a moving average.

3.3. Analysing UBN with a Toy Example

We illustrate the principles behind UBN and how it differs from BN using a toy example. We train a simple 5-layer fully connected network using BN layers on the Spiral Dataset[1], with 50 labeled examples and 400 unlabeled examples. Using this simple dataset allows us to do a deeper analysis which would not be straightforward using the depth estimation task.

We see from Fig. 2 that supervised training with BN learns the distribution of the labeled examples well and therefore learns accurate decision boundaries. However, those decision boundaries do not reflect the true data distribution. Instead, using UBN the network learns to align its decision boundary with the data manifold and learns to interpolate between the labeled examples by utilizing the batch statistics of unlabeled examples. In the coming sections we will examine the mechanism behind UBN through this example.

3.3.1 Bias in Batch Statistics

Batch statistics are calculated from labeled examples in BN. When there is only a limited number of labeled examples, this may cause the normalization statistics to be biased and end up hurting the model performance. We recorded the feature distributions from different layers of regularly spaced points, labeled points and unlabeled points while the network normalization statistics using the unlabeled examples. 3.

[1] http://playground.tensorflow.org

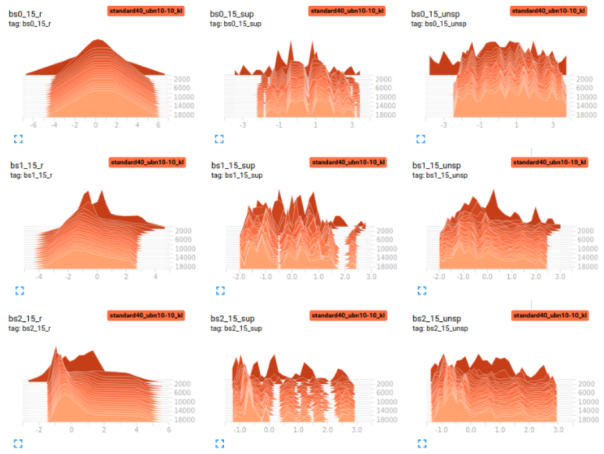

Figure 3. Histogram of hidden representations during training of the regularly sampled points (left column) the labeled points (middle column) and the unlabeled points (right column) after first (first row) second (second row) and third (third row) Batch Normalization layers of a random channel.

We have observed from the histogram of feature maps that the distribution of unlabeled and labeled examples are drastically different from each other. Calculating batch statistics with respect to only the labeled examples will cause a significant amount of bias in the normalization values. We postulate that the difficulty of obtaining correct batch statistics will be an important cause for the failure of Deep Learning methods in problems where the data is limited. Our method alleviates this problem by calculating the normalization values with respect to a larger variety of examples that better reflect the true data distribution.

3.3.2 Regularization in UBN

Our understanding is that the regularizing effect of UBN and similarly of BN comes from the noise that is induced by changing batch norm statistics which makes the network invariant to this type of noise. This specific noise is also different from simpler random variations that is applied to gradients [21] and feature maps [7] of the network since it comes from the actual data distribution and extends the data boundaries in the manifold that is predicted by the network. Other method for injecting learned noises are also not applicable in the semi-supervised setting where the small number of labels doesn't allow for such networks to be trained.

We visualized the uncertainty that is induced by changing batch statistics by making multiple predictions using the

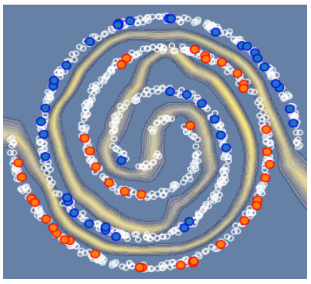

Figure 4. Uncertainty in the decision boundaries when unlabeled batch statistics are changed.

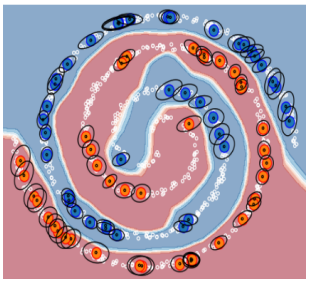

Figure 5. We visualize the data driven implicit augmentation via the noise that is induced by changing batch statistics.

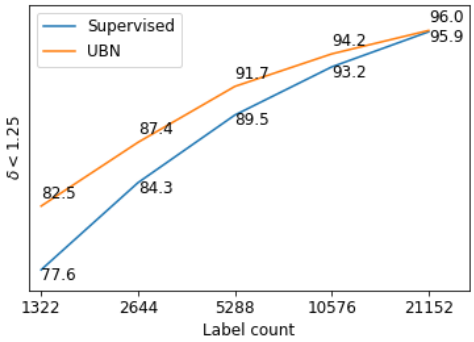

Figure 6. UBN shows a larger improvement as the number of labeled images decreases.

same inputs while updating the batch statistics [27]. This gives us a distribution of prediction from which we can calculate the standard deviation of our predictions. We observe that the uncertainty of the network follows the data manifold and aligns well with incorrect predictions and regions where there are no data examples 4.

We analyse the induced noise of changing batch statistics by making a forward pass of the network using labelled examples saving the network predictions updating the batch statistics and updating the inputs using back propagation in the feature space until the predictions are the same with the earlier batch statistics. This gives a distribution of starting points that have the same effect as changing the batch statistics. We observe that the augmentation follows the data manifold except for regions where the decision boundary is too close the the data manifold where in that case it pushes the decision boundary away from the labeled examples 5. Thus, Batch normalization layers apply an implicit augmentation that is dependant on the data manifold.

4. Experiments

KITTI dataset [8] contains stereo RGB images at 256x832 resolution and ground truth depth maps collected from Velodyne HDL-64E rotating 3D laser scanner at 10Hz. We have utilized the Kitti Raw dataset with 42304 training images and 2480 testing images.

The architecture for our method is based based on the SfMLearner architecture [33] which uses a modified version of the DispNet architecture [20]. It is a fully convolutional encoder-decoder architecture with skip connections and predictions in multiple resolutions. The building block of the architecture consists of 3x3 convolution operation, BN layers and a ReLU activation. The loss is calculated in different resolution by scaling the depth maps to match the prediction resolutions. The last layer before a prediction a sigmoid activation is used. The result of the sigmoid function is than scaled, shifted and inverted to obtain the depth prediction. The depth prediction D_i for a feature map X_i using a scaling factor of $s > 0$ and a minimum disparity value m such that $0 < m < 1$ can be calculated as follows

$$\hat{D}_i = \frac{1}{s * sigm(X_i) + m} \tag{5}$$

The network is trained with a learning rate of 0.0002 using the ADAM optimizer [14] for 160 epochs with batch size of 128. The scaling parameter s is 0.4 and the minimum disparity value m is 0.0125 (max depth 80m) throughout the experiments.

In order to evaluate the effect of unlabeled examples on the BN statistics we compare the performance of the depth prediction network on the KITTI dataset where a certain fraction of the depth maps are missing while we keep the number of images the same.

Removing depth annotation in this way let us keep the variation in the dataset stable while reducing the total number of annotated examples drastically. The missing depth maps are designed to mimic a depth sensor with lower frame rate.

For the metrics we used accuracy δ with threshold τ : $\delta = max(\frac{\hat{D}_i}{D_i}, \frac{D_i}{\hat{D}_i}) < \tau$ for $\tau = 1.25$.

UBN shows progressively larger improvement as the number of labeled examples shrink 6 and outperforms the supervised BN training. If the number of labeled images is close to the unlabeled images the two methods become effectively the same which makes our method more suitable for problems with limited labels.

5. Conclusion

We have shown that UBN reduces the bias in batch statistics and applies a regularization that utilizes the data manifold. Our method is easy to implement, computationally inexpensive and is effective in problems where obtaining annotations is difficult.

References

[1] A. J. Amiri, S. Y. Loo, and H. Zhang. Semi-supervised monocular depth estimation with left-right consistency using deep neural network, 2019. 2

[2] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 2

[3] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel. Mixmatch: A holistic approach to semi-supervised learning. *arXiv:1905.02249*, 2019. 2

[4] N. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger. Understanding batch normalization. In *Advances in Neural Information Processing Systems*, pages 7694–7705, 2018. 2

[5] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 2

[6] L. Deecke, I. Murray, and H. Bilen. Mode normalization. *arXiv preprint arXiv:1810.05466*, 2018. 2

[7] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016. 3

[8] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 4

[9] V. Guizilini, J. Li, R. Ambrus, S. Pillai, and A. Gaidon. Robust semi-supervised monocular depth estimation with reprojected distances, 2019. 2

[10] X. Guo, H. Li, S. Yi, J. Ren, and X. Wang. Learning monocular depth by distilling cross-domain stereo networks. *Lecture Notes in Computer Science*, page 506–523, 2018. 2

[11] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017. 2

[12] S. Ioffe. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. In *Advances in neural information processing systems*, pages 1945–1953, 2017. 1

[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 1, 2

[14] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[15] J. N. Kundu, P. K. Uppala, A. Pahuja, and R. V. Babu. Adadepth: Unsupervised content congruent adaptation for depth estimation, 2018. 2

[16] Y. Kuznietsov, J. Stückler, and B. Leibe. Semi-supervised deep learning for monocular depth map prediction, 2017. 2

[17] D.-H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2, 2013. 2

[18] P. Luo, X. Wang, W. Shao, and Z. Peng. Towards understanding regularization in batch normalization. *arXiv preprint arXiv:1809.00846*, 2018. 2

[19] Y. Luo, J. Ren, M. Lin, J. Pang, W. Sun, H. Li, and L. Lin. Single view stereo matching, 2018. 2

[20] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, 2016. 4

[21] A. Neelakantan, L. Vilnis, Q. V. Le, I. Sutskever, L. Kaiser, K. Kurach, and J. Martens. Adding gradient noise improves learning for very deep networks, 2015. 3

[22] T. Salimans and D. P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in neural information processing systems*, pages 901–909, 2016. 2

[23] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems*, pages 2483–2493, 2018. 2

[24] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018. 1, 2

[25] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019. 1

[26] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204, 2017. 2

[27] M. Teye, H. Azizpour, and K. Smith. Bayesian uncertainty estimation for batch normalized deep networks. *arXiv preprint arXiv:1802.06455*, 2018. 2, 4

[28] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 2

[29] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019. 2

[30] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 1

[31] A. R. Zamir, A. Sax, W. B. Shen, L. J. Guibas, J. Malik, and S. Savarese. Taskonomy: Disentangling task transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018. 2

[32] R. Zhang, P. Isola, and A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, pages 645–654, United States, 11 2017. Institute of Electrical and Electronics Engineers Inc. 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017 ; Conference date: 21-07-2017 Through 26-07-2017. 1

[33] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *Pro-*

ceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1851–1858, 2017. 4

[34] X. J. Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005. 1